

What is claimed is:

1 1. A method of managing a plurality of software development environments
2 coupled to one another through a cross development environment, the method
3 comprising:
4 detecting an update made in a first software development environment
5 among the plurality of software development environments; and
6 dynamically modifying a mapping between the first software development
7 environment and a second software development environment among the plurality
8 of software development environments in response to the detected update.

1 2. The method of claim 1, wherein each of the first and second software
2 development environments comprises a problem tracking tool, and wherein the mapping
3 is configured to convert a problem report generated by the problem tracking tool in the
4 first development environment to an acceptable format for the problem tracking tool in
5 the second development environment.

1 3. The method of claim 1, wherein the mapping is configured to convert a source
2 code file generated by the first development environment to an acceptable format for the
3 second development environment.

1 4. The method of claim 1, wherein detecting the update comprises receiving a
2 notification from the first software development environment.

1 5. The method of claim 1, further comprising notifying an administrator of the
2 cross development environment in response to detecting the update.

1 6. The method of claim 1, further comprising notifying an administrator of the
2 second software development environment in response to detecting the update.

1 7. The method of claim 6, further comprising updating the second software
2 development environment in response to notification of the administrator.

1 8. The method of claim 1, wherein the update comprises an update to content
2 stored in the first software development environment.

1 9. The method of claim 8, wherein the update comprises an update to at least one
2 of a product, component and release stored in a library repository in the first software
3 development environment.

1 10. The method of claim 1, wherein the update comprises an update to at least
2 one of a tool, a parameter and a value in the first software development environment.

1 11. The method of claim 1, wherein the mapping is defined in a mapping data
2 structure comprising a plurality of mapping entries, wherein at least one mapping entry
3 includes a wildcard.

1 12. The method of claim 1, further comprising transforming a transaction
2 generated by the first software development environment into a format compatible with
3 the second software development environment using the mapping.

1 13. The method of claim 12, wherein transforming the transaction includes
2 routing the transaction to one of a plurality of cross development environment processes.

1 14. The method of claim 13, wherein routing the transaction is performed by a
2 router process, the router process configured to perform at least one of failover and load
3 balancing in connection with routing the transaction to a cross development environment
4 process.

1 15. The method of claim 12, wherein transforming the transaction includes
2 communicating the transaction to the second software development environment, the
3 method further comprising retrying communication of the transaction to the second
4 software development environment in response to unavailability of the second software
5 development environment.

1 16. A method of managing a plurality of software development environments
2 coupled to one another through a cross development environment, the method
3 comprising:
4 updating a first software development environment among the plurality of
5 software development environments; and
6 notifying the cross development environment of the update made in the
7 first software development environment in response to the update.

1 17. The method of claim 16, further comprising dynamically modifying a
2 mapping between the first software development environment and a second software
3 development environment among the plurality of software development environments in
4 response to notification of the cross development environment.

1 18. The method of claim 16, further comprising dynamically notifying an
2 administrator in response to the update.

1 19. An apparatus, comprising:

2 a memory configured to store a mapping data structure for use in a cross
3 development environment that couples together a plurality of software
4 development environments;

5 a processor; and

6 program code configured to detect an update made in a first software
7 development environment among the plurality of software development
8 environments, and dynamically modify the mapping data structure to modify a
9 mapping between the first software development environment and a second
10 software development environment among the plurality of software development
11 environments in response to the detected update.

1 20. The apparatus of claim 19, wherein each of the first and second software
2 development environments comprises a problem tracking tool, and wherein the mapping
3 data structure is configured to convert a problem report generated by the problem tracking
4 tool in the first development environment to an acceptable format for the problem
5 tracking tool in the second development environment.

1 21. The apparatus of claim 19, wherein the mapping data structure is configured
2 to convert a source code file generated by the first development environment to an
3 acceptable format for the second development environment.

1 22. The apparatus of claim 19, wherein the program code is configured to detect
2 the update by receiving a notification from the first software development environment.

1 23. The apparatus of claim 19, wherein the program code is further configured to
2 notify an administrator of the cross development environment in response to detecting the
3 update.

1 24. The apparatus of claim 19, wherein the program code is further configured to
2 notify an administrator of the second software development environment in response to
3 detecting the update.

1 25. The apparatus of claim 19, wherein the update comprises an update to content
2 stored in the first software development environment.

1 26. The apparatus of claim 25, wherein the update comprises an update to at least
2 one of a product, component and release stored in a library repository in the first software
3 development environment.

1 27. The apparatus of claim 19, wherein the update comprises an update to at least
2 one of a tool, a parameter and a value in the first software development environment.

1 28. The apparatus of claim 19, wherein the mapping data structure includes a
2 plurality of mapping entries, wherein at least one mapping entry includes a wildcard.

1 29. The apparatus of claim 19, wherein the program code is further configured to
2 transform a transaction generated by the first software development environment into a
3 format compatible with the second software development environment using the mapping
4 data structure.

1 30. The apparatus of claim 29, further comprising a plurality of cross
2 development environment processes and a router process configured to receive the
3 transaction and route the transaction to one of the cross development environment
4 processes.

1 31. The apparatus of claim 30, wherein the router process is configured to
2 perform at least one of failover and load balancing in connection with routing the
3 transaction to a cross development environment process.

1 32. The apparatus of claim 29, wherein the program code is further configured to
2 communicate the transformed transaction to the second software development
3 environment, and to retry communication of the transaction to the second software
4 development environment in response to unavailability of the second software
5 development environment.

1 33. An apparatus, comprising:

2 a memory configured to store a mapping data structure for use in a cross
3 development environment that couples together a plurality of software
4 development environments, wherein the mapping data structure includes at least
5 one wildcarded field;

6 a processor; and

7 program code configured to transform a transaction generated by a first
8 software development environment among the plurality of software development
9 environments into a format compatible with a second software development
10 environment among the plurality of software development environments using the
11 wildcarded field in the mapping data structure.

1 34. A computer system comprising:

2 first and second software development environments coupled to one
3 another by a cross development environment; and

4 program code resident in the first software development environment and
5 configured to notify the cross development environment of an update made in the
6 first software development environment.

1 35. The computer system of claim 34, further comprising program code resident
2 in the cross development environment and configured to dynamically modify a mapping
3 between the first software development environment and the second software
4 development environment in response to notification of the cross development
5 environment.

1 36. The computer system of claim 34, further comprising program code
2 configured to dynamically notify an administrator in response to the update.

1 37. A program product, comprising:

2 program code configured to detect an update made in a first software
3 development environment among a plurality of software development
4 environments coupled to one another by a cross development environment, and
5 dynamically modify a mapping between the first software development
6 environment and a second software development environment among the plurality
7 of software development environments in response to the detected update; and
8 a computer readable signal bearing medium bearing the program code.

1 38. The program product of claim 37, wherein the signal bearing medium
2 includes at least one of a recordable medium and a transmission medium.